

Reusable Asset Specification Version 2.2 Final



Eric Nelson, Industry Advisory Team

February 21, 2006



Scope of the RAS

- The scope of this Specification is a **set of guidelines and recommendations about the structure, content, and descriptions of reusable software assets**. We recognize that there are different categories of reusable software assets. The specification identifies some categories, or rather types or profiles and provides general guidelines on these profiles.
- The Reusable Asset Specification (RAS) **addresses the engineering elements of reuse**. It attempts to reduce the friction associated with reuse transactions through consistent, standard packaging. This is much like the steering wheel, turn signals, pedals, and fuel gauge in a car: although they're slightly different across car models and makes, there's a familiarity among them that significantly reduces the costs of reuse.

How do you package an asset to support reuse?



RAS Provides

- Extensible specification of [software] reusable assets
 - Profiles to extend a Core spec to categories of assets
 - Component, Web Services
- Standard data interchange for tool vendors
 - XML mapping for tool interchange

■ <http://www.omg.org/cgi-bin/apps/doc?ptc/05-04-02.pdf>



Definition: Reusable Asset

- “...provide[s] a solution to a problem **for a given context.**”
 - <1..*> Artifacts
 - Rules for usage
 - [optional] Variability Points
- An Asset Package is
 - Collection of artifacts and a manifest



Aspects of a Reusable Asset

- Granularity
 - How many problems does it solve?
- Variability
 - How much can it be changed?
 - [clear | white | grey | black] box
- Articulation
 - How much of the solution is provided vs. specified?
















What is an Artifact?

- A file:
 - Documentation
 - Requirements, Design, Testing
 - Code
 - Source, binaries, etc
 - Makefiles, dependency files, etc.
 - Models
 - Configuration Files
 - Test Scripts
 - Etc.

Asset Packaging (Archive)

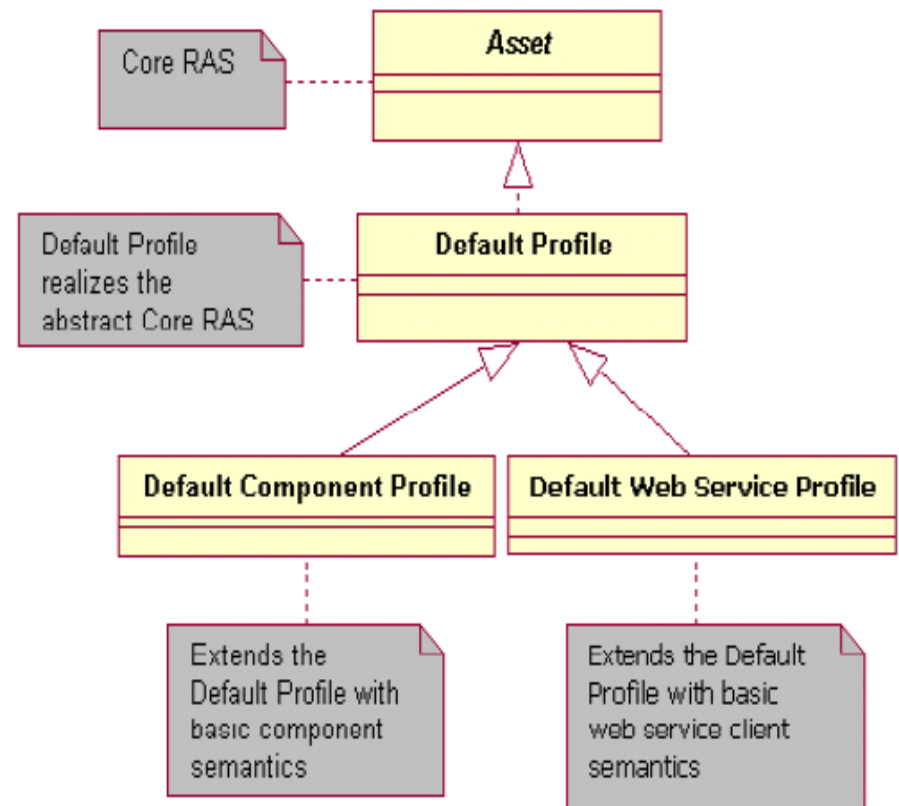
- Required
 - Asset Manifest
 - Profile XML Schema
 - 1..* Artifacts
- Packaging types
 - “Bundled” – zip file in an asset repository to be distributed by file exchange
 - “Unbundled” – manifest points to artifacts “in place,” e.g. file systems, or version control repositories

rasset.xml
is the
manifest
and points
to the other
files

 java code model.mdx	749 KB
 RASDefault/WebServiceProfile.xsd	19 KB
 rasset.xml	5 KB
 servicebindingexample.java	3 KB
 serviceproperties.xml	1 KB
 sqs.wsdl	1 KB
 sqs-interface.wsdl	2 KB
 stockquotedemo.bat	1 KB
 stockquotedemo.sh	1 KB
 supplementaryspecification.doc	56 KB
 usecasemodel.mdx	744 KB
 usecasemodel.wdx	1 KB
 webserviceusecase.gif	2 KB

Core RAS and Profiles

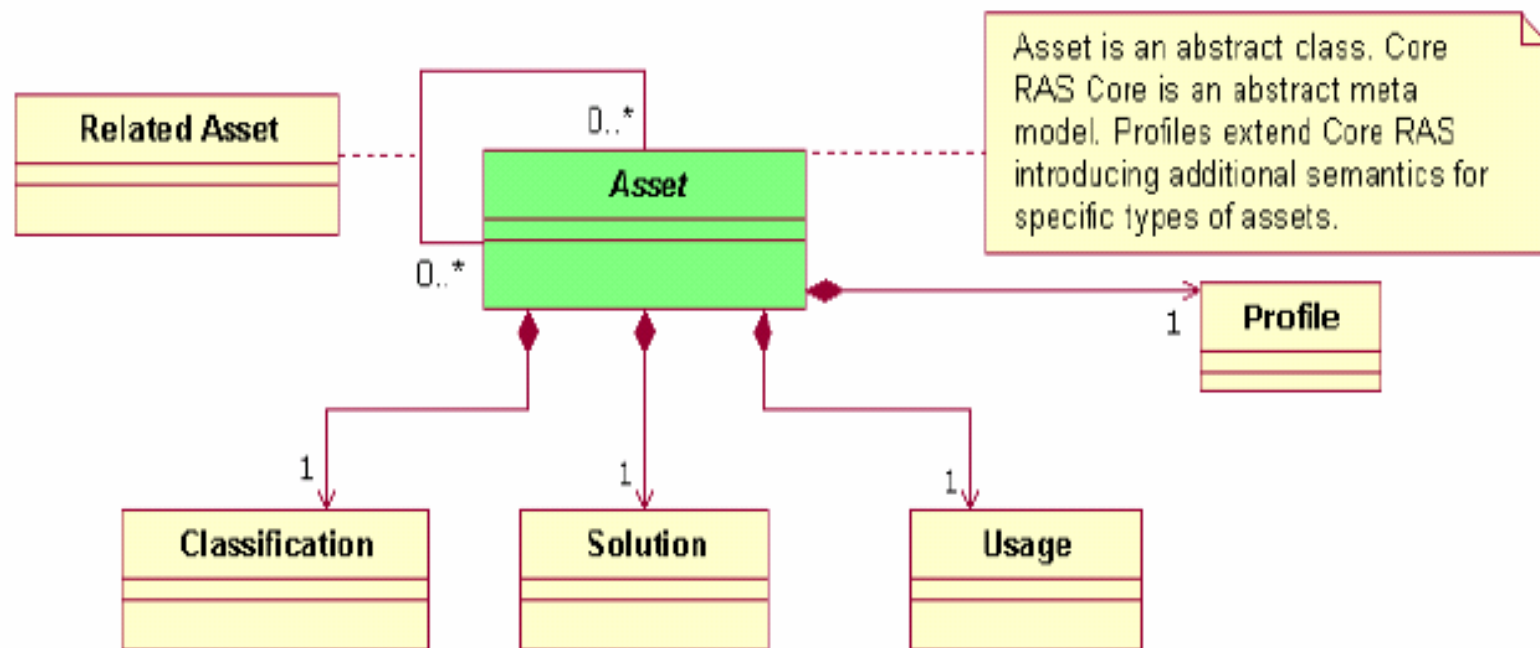
- Core RAS is an abstract element
 - Each profile adds semantics that support an asset “type”
- Profiles in the RAS Spec
 - Default (minimal)
 - Component
 - Web Service





Core RAS Elements

- Must be part of any profile





Profiles

- Profiles extend the CoreRAS model
 - Must reference the parent profile
 - Add relations and attributes
 - Additional semantic constraints on elements, relations and attributes
 - Note: inherited constraints cannot be loosened
- Profiles included in the specification
 - Default (minimal semantics)
 - Default Component
 - Default Web Service



Example Profile: Default Component 1.1

■ Additions to Core

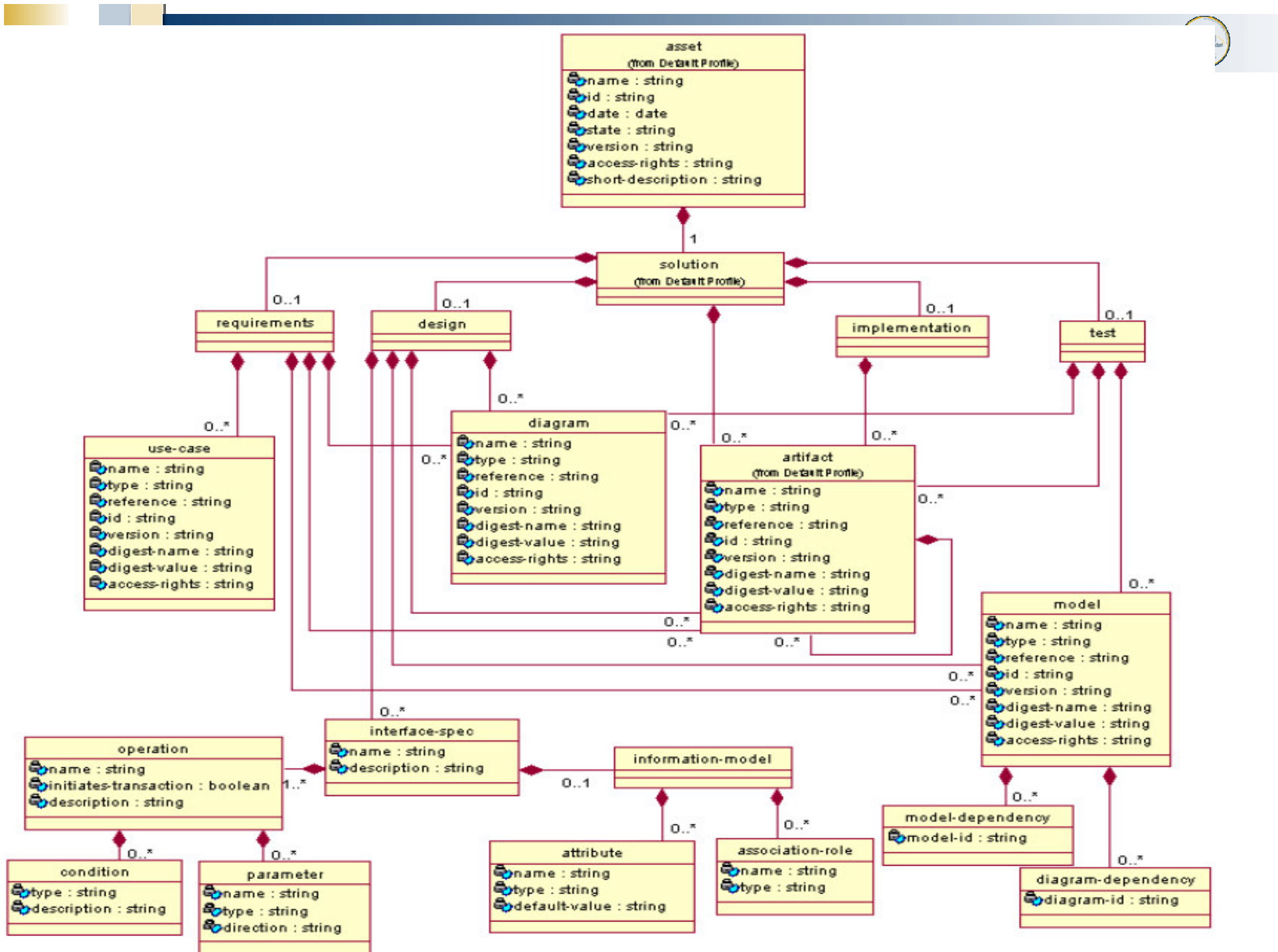
- Requirements
- Design
- Model
- Diagram
- Implementation
- Test
- Use-Case
- Interface-Spec
- Information-Model
 - Attribute
 - Association Role

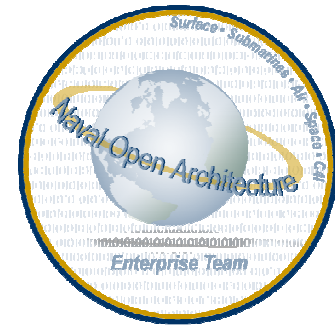
□ Operation

- Condition
- Parameter

■ Extensions to Core

- Solution, added relations to
 - Requirements
 - Design
 - Implementation
 - Test





Backup Slides



